




A novel and efficient 8-point DCT approximation for image compression

Nabila Brahimi¹  · Toufik Bouden² · Tahar Brahimi³ · Larbi Boubchir⁴

Received: 29 December 2018 / Revised: 17 September 2019 / Accepted: 1 October 2019

Published online: 02 January 2020

© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract

The Discrete Cosine Transform is widely used in the field of still image compression. Many integer approximations are given in the literature whereas the most of these transforms requires bit shift operations. This paper presents an efficient and low complexity integer approximation of the DCT for image compression. Our new approach involves replacing the bit shift elements of a variant of the Signed DCT transform by zeros, in order to eliminate the bit shift operations. As a result, all elements of the proposed transform are zeros and ± 1 . Indeed, the proposed transform retains all the characteristics of its original transform, such as orthogonality and high energy compaction capabilities, while generating computing cost savings. Experiments show that the proposed transform has a good compromise performance-computational complexity as well as state-of-the-art DCT approximations. Moreover, an efficient algorithm primarily involving a small amount of arithmetical computation is well developed as no multiplications and bit-shift operations are required, with only 16 additions being involved.

Keywords Image compression · DCT · Approximation · Fast algorithm · JPEG

1 Introduction

Data compression schemes aim to reduce the amount of data required to represent any digital data including images and signals [7–9]. Indeed, data compression is the art of information

✉ Nabila Brahimi
nabila.brahimi@univ-jijel.dz

Toufik Bouden
bouden_toufik@univ-jijel.dz

Tahar Brahimi
t.brahimi@gmail.com

Larbi Boubchir
boubchir@ai.univ-paris8.fr

Extended author information available on the last page of the article

representation in a compact form [24]. Transform coding is a well-established and commonly used technique for image compression. The main objective of the transformation is to remove redundancy and provide uncorrelated coefficients [7–9].

The discrete cosine transform (DCT) has originated in Ahmed et al. work [1] and has been used and studied extensively ever since. It is well-known for its highly coding performance and it is widely used in many image compression applications [18, 22, 28]. DCT processes a high energy compaction property, which is superior to any known transform with the fast-computational algorithm. It is this property that makes the DCT the most appropriate transform for lossy data compression. Various 8×8 discrete transform families have been advanced along with the fast algorithms for their computations. Among these transforms, the Signed DCT (SDCT) is interesting because it keeps the properties of the conventional DCT transform such as the good de-correlation and power compaction capabilities, and it can easily be obtained from the DCT [22]. Even though, the SDCT requires 24 additions but its performance in image compression is still very low. With the aim of improving the performance of the SDCT, several approximate DCT transforms have been proposed in the published literature. The series of Bas (Bouguezel, Ahmed and Swamy) transforms such as the parametric transform in [4] (Bas-2011) is obtained by introducing an arbitrary parameter in the transform matrix reported in [3] and performing some row permutations. The resulting transforms require 18 additions and 2 shifts, 18 additions or 16 additions depending on whether the parameter ‘ a ’ is equal to 2, 1 or 0 respectively. The best performance for these three transforms is obtained for $a = 2$, but it is still less than the one reported in [3] even they have the same number of operations.

Two other transforms with only 14 additions are also described in [11, 21], although their performance in image compression is rather poor. On the other hand, the authors in [13] have proposed an approach for the development of an 8×8 integer DCT based on higher-dimensional DCT (16×16), of which the three transforms obtained have good image compression performance. However, they handle a growing computational complexity, (they perform between 26 and 30 operations). Moreover, others transform are equally developed in [2, 5, 6, 26].

The main contributions of this paper are as follows:

- Development of a new and fast orthogonal approximation for the 8-point DCT, based on the approximate 8-point transform introduced in [3], by eliminating the two bit-shift operations performed in [15] and offering a trade-off between image quality and the number of operations involved. Compared to [11], the proposed transform obtains superior performance advantages. Note that the bit-shift operations cause computational problems on the hardware implementation [15];
- Derivation of a fast algorithm for the proposed transform;
- An objective assessment of the new approximation in terms of image compression performance compared to popular alternatives.

2 Exact DCT and its approximations

The discrete cosine transform is one of the techniques for converting the temporal signal into its frequency components. This technique is widely used in different areas of information processing, especially in image compression. The Standards: JPEG of Joint picture Expert

Group adopted in 1992 (ISO/IEC International Standard 10918-1 ITU-T Recommendation T. 81), MPEG of Moving Picture Expert Group developed by ISO (International Organization for Standardization) and H. 261, H. 263, H. 264 [22] proposed by ITU (International Telecommunication Union), use the DCT technique which is very close to discrete Fourier transform.

2.1 Exact DCT transform

Literature is very rich in books that explain the concept of the DCT, its theory, its fast algorithms and its integer approximations [23].

To get the i^{th} and j^{th} DCT transformed elements, noted $F(i, j)$, of an original image block, noted $f(i, j)$, of size $N \times N$, the following equation is used:

$$F(i, j) = \frac{1}{\sqrt{2N}} \alpha(i) \alpha(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \cos \left[\frac{\pi(2x+1)i}{2N} \right] \cos \left[\frac{\pi(2y+1)j}{2N} \right] \quad (1)$$

where $i, j = 0, 1, 2, 3, \dots, N-1$ are spatial frequency indices in the horizontal and vertical direction of the image block, $\alpha(i)$ and $\alpha(j)$ are given by:

$$\alpha(k) = \begin{cases} 1/\sqrt{2}, & k = 0 \\ 1 & \text{otherwise} \end{cases} \quad (2)$$

The inverse transform (IDCT) is given by:

$$f(x, y) = \frac{1}{\sqrt{2N}} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \alpha(i) \alpha(j) F(i, j) \cos \left[\frac{\pi(2x+1)i}{2N} \right] \cos \left[\frac{\pi(2y+1)j}{2N} \right] \quad (3)$$

where $x, y = 0, 1, 2, 3, \dots, N-1$.

For the quick calculation, we use the DCT matrix form whose elements are given by the formula:

$$T_{DCT} = \begin{cases} 1/\sqrt{N} & i = 0 \\ \sqrt{\frac{2}{N}} \cos \left[\frac{\pi(2j+1)i}{2N} \right] & \text{otherwise} \end{cases} \quad (4)$$

The discrete cosine transform has the advantage of being implemented by different fast algorithms.

2.2 Approximate DCT transforms

Image, video and audio digital signals are represented by integer values. The floating-point DCT converts these integer values to real coefficients. Although the fast algorithms of the DCT such as [5, 6, 12, 23, 27], reduce significantly the number of their arithmetical operations and required floating-point operations. They still make hardware and software implementations very slow, require a lot of memory space and consume too much electrical energy.

To remedy these problems, the DCT coefficients are approximated by integers, so the floating-point multiplication is replaced by integer multiplication [2–6, 11, 14, 15, 25, 26].

1.1.1. PADCT transform

Principle in [15] is simple, it consists of introducing the signum function in the DCT matrix transform. Then the resulting transform, PDCT, is quasi orthogonal as its original transform; two elements in $\mathbf{T}_1 \times \mathbf{T}_1^t$ are different to zero (noted that “t” is the transpose operation). The complexity of this transform is 17 additions and it is given by:

$$\mathbf{C}_1 = \mathbf{D}_1 \times \mathbf{T}_1 \quad (5)$$

$$\mathbf{T}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (6)$$

$$\mathbf{D}_1 = 1/\sqrt{\mathbf{T}_1 \mathbf{T}_1^t} = \text{diag} \left(1/\sqrt{8}, 1/2, 1/\sqrt{8}, 1/\sqrt{2}, 1/\sqrt{8}, 1/2, 1/2, 1/\sqrt{2} \right) \quad (7)$$

2.2.1 Parametric transform

Bouguezet et al. proposed a new approximate DCT transform in [4], who they called parametric transform. It consists of introducing an arbitrary parameter a , in the transform reported in [3] and performing some row permutation. Its orthogonality is verified in [4].

$$\mathbf{C}_2 = \mathbf{D}_2 \times \mathbf{T}_2 \quad (8)$$

$$\mathbf{T}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & a & -a & -1 & -1 & -a & a & 1 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \\ a & -1 & 1 & -a & -a & 1 & -1 & a \end{bmatrix} \quad (9)$$

$$\mathbf{D}_2 = \text{diag} \left(1/\sqrt{8}, 1/\sqrt{4}, 1/\sqrt{4+4a^2}, 1/\sqrt{2}, 1/\sqrt{8}, 1/\sqrt{2}, 1/\sqrt{4}, 1/\sqrt{4+4a^2} \right) \quad (10)$$

A fast algorithm is developed in [4] to reduce the number of arithmetical operations from 36 additions and 8 multiplications to few additions and bit shift operations. For example:

- $a = 0$, T_2 required 16 additions.
- $a = 1$, T_2 required 18 additions.
- $a = 2$, T_2 required 18 additions and 2 bit-shift.

2.2.2 Signed DCT

The principle given in [14] consists of applying the signum function to the DCT matrix elements. The resulting transform is named SignDCT (SDCT), noted by, T_N^{SDCT} , and given by:

$$T_N^{SDCT}(i, j) = \frac{1}{\sqrt{N}} \text{sign}\{T_N(i, j)\} \quad (11)$$

where “sign” means the signum function, defined by:

$$\text{sign}\{x\} = \begin{cases} +1 & \text{if } x > 0, \\ 0 & \text{if } x = 0, \\ -1 & \text{if } x < 0. \end{cases} \quad (12)$$

The application of the signum function on the DCT matrix gives this simple matrix:

$$T_3 = \frac{1}{\sqrt{8}} \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & -1 & +1 & +1 & +1 & -1 \\ +1 & -1 & -1 & +1 & +1 & -1 & -1 & +1 \\ +1 & -1 & +1 & +1 & -1 & -1 & +1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \end{bmatrix} \quad (13)$$

It can be verified that the transform matrix T_3 is not orthogonal, i.e. the number of non-zero off-diagonal elements of $T_3 \times T_3^t$ is 12%. So different fast algorithms are used to calculate the forward and inverse SDCT transformations.

3 Proposed transform

The paper aims to develop a fast and efficient integer approximate DCT transform by performing a good trade-off between energy compaction capabilities and the number of arithmetic operations required. As mentioned earlier in the introduction section, several approaches have been proposed while the majority of these transforms, producing good image compression performance, require bit shift operations.

Indeed, the bit-shift operations cause computational problems on the hardware implementation [15]. To remedy these problems, a new approximation has been proposed in [15] resulting from the application of the signum function to the transformation given in [25]. It only requires additions; however, it is quasi-orthogonal and its image compression performance is quite low. The transform in [3] is a variant of

SDCT [14], which consists of introducing some 0 and $\pm 1/2$ elements. The resulting transform requires 18 additions and 2 bits shift operations. To avoid such problems, we propose a different variant of [3] completely free of bit shift and multiplication operations by replacing the elements equal to $1/2$ of the matrix transform in [3] with 0.

The transform matrix given in [3], \mathbf{T} , and the associated proposed transform, referred to as \mathbf{T}_p , with its diagonal \mathbf{D}_p are given by:

$$\mathbf{T} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 1/2 & -1/2 & -1 & -1 & -1/2 & 1/2 & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \\ 1/2 & -1 & 1 & 1/2 & -1/2 & 1 & -1 & 1/2 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (14)$$

$$\mathbf{T}_p = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & -1 & 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (15)$$

$$\mathbf{D}_p = \text{diag}\left(1/\sqrt{8}, 1/2, 1/2, 1/\sqrt{2}, 1/\sqrt{8}, 1/2, 1/2, 1/\sqrt{2}\right) \quad (16)$$

$$\mathbf{C}_p = \mathbf{D}_p \times \mathbf{T}_p$$

All elements of our matrix are 0 or ± 1 , so multiplication and bit-shift operations are entirely absent. It is easy also to verify that our new transform is orthogonal, i.e. $\mathbf{C}_p^{-1} = \mathbf{C}_p^t = \mathbf{T}_p^t \times \mathbf{D}_p$ where \mathbf{t} denotes the matrix transpose operation, unlike the one proposed in [15]. Hence, the same number of operations is required to compute both the forward and inverse transformations.

It is clear from Eq. (15) that the proposed transform requires 27 additions. To make our transform more obvious, we propose a fast algorithm for its calculation. This algorithm, based on sparse matrix factorization [19, p:121], consists of factorizing the matrix \mathbf{T}_p into three simple matrices product. Accordingly, the number of required operations is reduced to 16 additions only. The three matrices are given as follows:

$$\mathbf{T_P} = \mathbf{G_3} \times \mathbf{G_2} \times \mathbf{G_1} \quad (17)$$

$$\mathbf{G_1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (18)$$

$$\mathbf{G_2} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (19)$$

$$\mathbf{G_3} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (20)$$

The structure corresponding to our fast algorithm is illustrated in Fig. 1.

Figure 2 illustrates the basis element building bloc of our algorithm. Symbol equation effort 2 additions.

As Fig. 1 clearly illustrates, our transform requires only 16 additions. The structure can be divided into 3 independent stages. From left to right, the first stage requires 8 additions, the

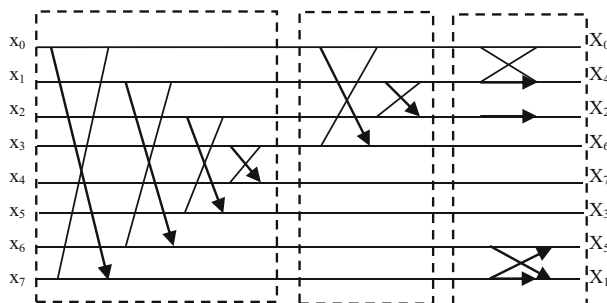


Fig. 1 Graphic illustration of the signal flow for the proposed fast algorithm using 8 points

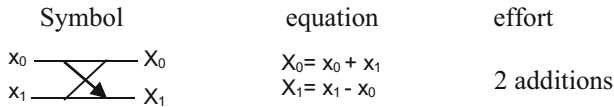


Fig. 2 Basis element used to display the fast-proposed algorithm

second one 4 additions, and finally the third 4 additions. To highlight our transformation, we compared it with other existing transformations in the literature in terms of complexity. Table 1 summarises the number of operations required for each transform.

The Scaled DCT is the fastest algorithm for calculating the DCT transform, and it is used only as a reference for performance assessment. From Table 1, we can show a reduction of 4 arithmetic operations compared to its original transform in [3]. Indeed, the transform we propose, with only 16 additions, also allows a significant reduction in the number of operations required compared to other transforms; it has 33.34%, 20%, 11.11% and 5.9% lower arithmetic costs than the SDCT [14], Bas ($a = 2$) [4], Bas ($a = 1$) [4], and PADCT [15], respectively. The transform with 14 additions in [11] remains the best in terms of complexity; however, it shows a significant degradation in terms of performance.

It is known that the good approximation of DCT transform must satisfy the following conditions:

- To have a low computational complexity, the most important requirement;
- The error energy of the approximation must be small in order to provide a compression performance close to the exact DCT;
- It is preferred that the approximation be orthogonal;
- Approximation must be able to be expanded to larger sizes to support modern video encoding standards.

The proposed transformation provides a compromise between the first three requirements above, the latter can be obtained by using the scalable recursive algorithm proposed in [16]. This algorithm has resulted in more dimensional versions of the proposed T_p matrix that are suitable for the video experiments. Note that this algorithm is already used in a more recent work published in [20].

Table 1 Comparative analysis of various DCT approximation methods in terms of computational complexity

Transform	Additions	shifts	Multiplications	Total
Proposed transform	16	0	0	16
Bas-2008 [3]	18	02	0	20
PADCT [15]	17	0	0	17
Bas-2011($a = 2$) [4]	18	2	0	20
Bas-2011($a = 1$) [4]	18	0	0	18
Bas-2011($a = 0$) [4]	16	0	0	16
Bas-2009 [2]	21	3	0	24
CB-2012 [11]	14	0	0	14
SDCT [14]	24	0	0	24
Scaled DCT [27]	29	0	5	34

4 Application in image compression

The input image is divided in 8×8 blocks. If we consider \mathbf{X} an input 8×8 block and \mathbf{F} the corresponding block in the transformed domain, the forward and inverse transformations are achieved respectively as follows:

$$\mathbf{F} = \mathbf{TX}\mathbf{T}^t = \mathbf{D}(\mathbf{CXC}')\mathbf{D} \quad (21)$$

$$\mathbf{X} = \mathbf{T}^t\mathbf{FT} = \mathbf{C}^t(\mathbf{DFD})\mathbf{C} \quad (22)$$

It should be noted that the diagonal matrix \mathbf{D} can be involved in the quantization/dequantization matrix. Consequently, the proposed matrix \mathbf{T}_p is the only factor taking part in arithmetical operations in the transformation stage of the image compression process.

5 Experimental validation

To demonstrate the efficiency of the proposed transform compared to its existing approximate DCT transforms counterparts, the following six well-known grayscale test images (8 bits per pixel): Lena, Barbara, Boat, Baboon, Sailboat and Airplane of size 512×512 will be considered. For this purpose, we carry out an experiment similar to that in [14] which is summarised in the following points:

- Each test image is divided into sub-blocks of size 8×8 ;
- The two-dimensional process transformation of all considered transforms: the Scaled DCT matrix in [27], SDCT matrix in [14], PADCT in [15], Bas-2011 with $a = \{2, 1, 0\}$ in [4], CB-2012 in [11] and the proposed transform matrix is applied to each sub-block. As a results 64 transformed coefficients is obtained for each sub-block. Note that the Scaled DCT [27] is employed only for the sake of reference;
- The standard zigzag lecture is applied and only the 35 initial coefficients in each block were employed to reconstruct the image. All the remaining coefficients were set to zero.

The performance of the proposed approximate DCT is assessed in terms of energy compaction characteristics compared to the DCT approximations suggested above. To this end, the three evaluation criteria, PSNR (Peak Signal to Noise Ratio), PEEN (Percentage Error Energy Norm) [14] and SSIM (Structural SIMilarity index) are chosen as they are the most used metrics for image distortion.

- The PSNR in our case is calculated from the Mean Squared Error (MSE) as follows:

$$\text{MSE} = \frac{1}{N \cdot N} \sum_{m=1}^N \sum_{n=1}^N (f(m, n) - \hat{f}(m, n))^2 \quad (23)$$

$$\text{PSNR} = 10 \log_{10} \frac{(255)^2}{\text{MSE}} \text{dB} \quad (24)$$

- The PEEN, which is a measure of the distortion (or relative distance) between the original image and the reconstructed image after the decompression process, is given as:

$$\text{PEEN} = \sqrt{\frac{\sum_{m=1}^N \sum_{n=1}^N \left(\frac{\mathbf{f}(m, n) - \hat{\mathbf{f}}(m, n))^2}{\sum_{m=1}^N \sum_{n=1}^N \mathbf{f}^2(m, n)} \times 100 \right)}{2}} \quad (25)$$

- The Structural SIMilarity index [29] is a full reference metric, in other words, the measuring of image quality based on an initial uncompressed or distortion-free image as reference. The measure between two windows of size $N \times N$ is:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\text{cov}_{xy} + c_2)}{(y_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (26)$$

where μ_x is the average of x ; μ_y is the average of y ; σ_x the variance of x ; σ_y the variance of y ; cov_{xy} the covariance between x and y ; $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$ two variables to stabilise the division with weak denominator; L the dynamic range of the pixel values.; $k_1 = 0.01$ and $k_2 = 0.03$ by default.

\mathbf{f} and $\hat{\mathbf{f}}$ in (23) are the $N \times N$ matrices of the original and reconstructed images, respectively.

Figure 1 shows the plot of average PSNR, average PEEN and average SSIM values obtained for the test images using different approximation methods mentioned above.

Figure 1a indicates that the proposed transform and Bas-2011 $\{a = 2\}$ [4] achieve similar results although our transform retains 11.11% of the number of the required operations. Compared with the same transform with $\{a = 0\}$, our transform shows a slight improvement at middle bit rates (with the number of the coefficients retained ranging from 20 to 27). However, we can notice a significant improvement in performance at high bit rates. Compared to the remaining transforms, the one we propose provides better values of PSNR at all compression ratios, even though it illustrates the lowest complexity. Note that the transform CB-2012 [11] required 14 additions only. Nevertheless, we can notice a decline in performance.

In Fig. 3b, we have shown the plot of average PEEN values obtained for different approximation transforms. As can be seen from Fig. 3b, the proposed algorithm delivers the best performance in comparison with the most existing methods at all compression ratios. The Fig. 3c confirms the results discussed above.

The difference in terms of PSNR between the proposed transform and the parametric transform in [4], for $a = \{2, 1, 0\}$, is illustrated in Fig. 4a. As indicated by the above curves, the overall performance of our transform shows an improvement as compared to the previous one. In comparison with the transform in [4] ($a = 2$), requiring 18 additions and 2 bit-shifts, the proposed transform, with just 16 additions, provides significant improvements in performance in terms of both PSNR and computational complexity, as is evident from Fig. 4a and Table 1, respectively. With respect to the transform in [4] with $a = 1$ and 18 additions, our transform

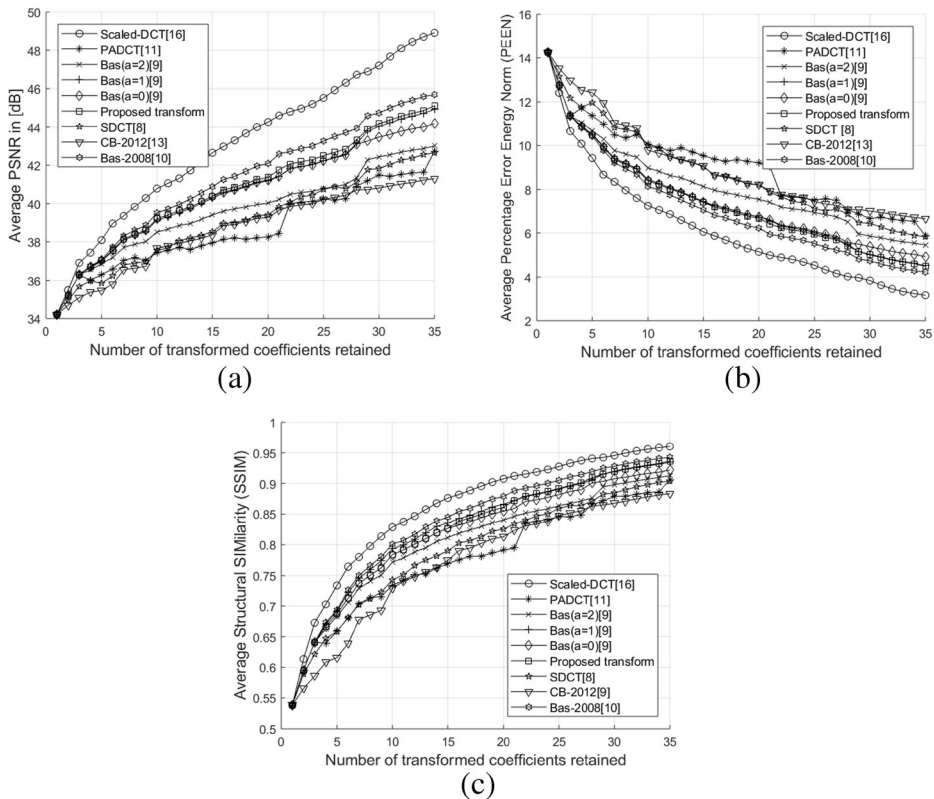


Fig. 3 Quality assessment for different compression ratios (a) Average PSNR (b) Average PEEN (c) Average SSIM

produces a slight improvement, with only 16 additions. For $a = 0$, without increasing complexity, similar results are displayed at low bitrates (the number of coefficients retained is less than 15). On the other hand, at high bit rates (the number of transformed coefficients retained is greater than 30), our transformation seems to be better.

Figure 4b shows the difference between the PSNR obtained by our proposed transform and that obtained by the transform in [15]. It is clear from this Figure that along with the number of additions reduced by one addition, another still more important improvement in PSNR is achieved by our transform.

The difference between our transform and its original transform [3] is shown in Fig. 4c. A slight degradation can be observed at the high compression ratio; this degradation is increased when the compression ratio is decreased. And that's due to fact that our transform presents a reduction of 4 arithmetical operations compared to its original transform.

Table 2 summarises some numerical values of the average PSNR resulting from the application of each transform on the test images for different numbers of coefficients retained for the image reconstruction. The PSNR values of the best transform that provides a good compromise between the number of operations and the compression performance are highlighted, as shown in Tables 2 and 4. These results provide clear evidence of an improvement in performance by using our transform, in addition to performing a small number of

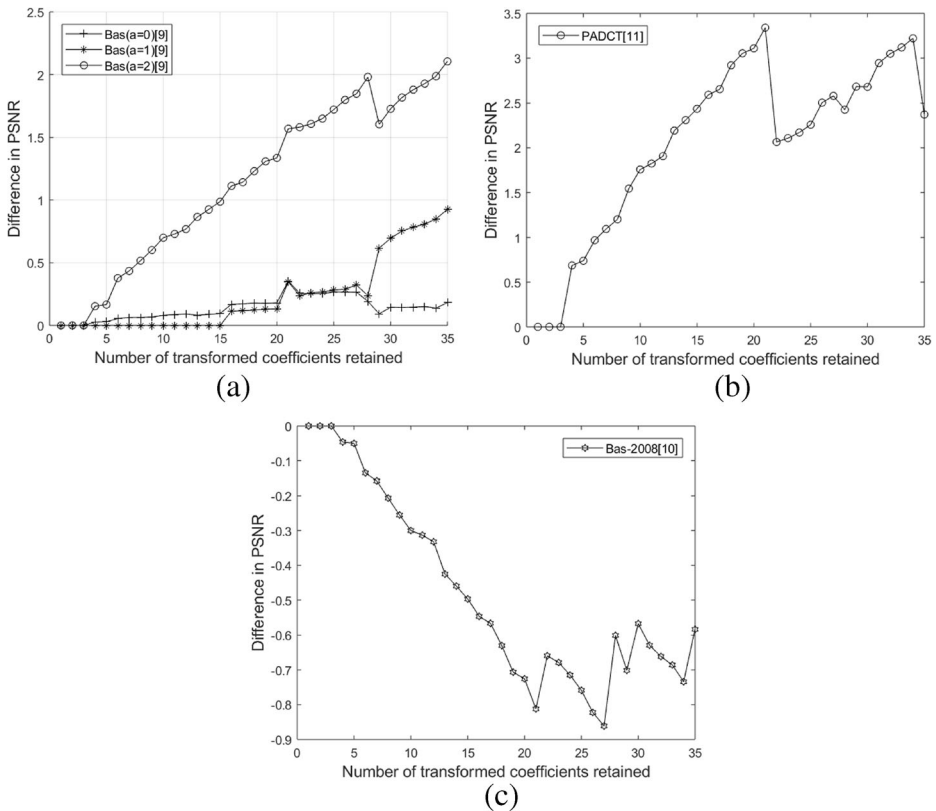


Fig. 4 The gain in PSNR of the proposed transform over (a) Transform in [4] with $a = \{0, 1, 2\}$ (b) Transform in [15] (c) Transform in [3]

operations. As compared to the transform in [3], requiring 18 additions and 2-bit shifts operations, the one we proposed shows a slight decrease in PSNR. In contrast, it generates cost savings in computation, without bit shift operations and with only 16 additions.

6 Complexity analysis in terms of comparing the running time of the forward transformation

The running time comparison between the proposed fast algorithm and that given in [3] is carried out using the following platform (Intel Core I5-8250U Processor $\times 64$ at 1.6 GHz, 8 GB memory) using MATLAB programming language. Table 3 shows the average values obtained for performing the forward transform. For that, three Lena greyscale images of size 256×256 , 512×512 , 1024×1024 , are used.

As can be seen in Table 3, the result confirms what was shown in our previous study when comparing the arithmetic operations. Indeed, the comparison of the running times shows that the proposed algorithm is even better. This is due to the fact that our transform performs a lowest computation.

Table 2 Average PSNR (in dB) of different transforms

Methods	Number of arithmetic operations	Number of coefficients retained			
		05	10	20	30
Proposed transform	16 additions	37.03	39.20	41.36	44.17
Bas-2008 [3]	18 additions and 2 bit shifts	37.08	39.50	42.09	44.73
PADCT [15]	17 additions	36.29	37.45	38.26	41.49
Bas-2011 [4] (a = 2)	18 additions and 2 bit shifts	36.86	38.50	40.03	42.44
Bas-2011 [4] (a = 1)	18 additions	37.00	39.12	41.18	44.02
Bas-2011 [4] (a = 0)	16 additions	37.03	39.20	41.23	43.47
CB-2012 [11]	14 additions	35.85	37.22	39.29	40.42
SDCT [14]	24 additions	35.34	37.56	39.40	41.83
Scaled DCT [27]	29 additions and 5 multiplications	38.08	40.78	44.26	47.19

The Table 4 summarises the results obtained when implementing several approximate DCT transforms under the JPEG compression standard.

First, we compare the compression performance of the proposed transform with that of the original transform proposed in [3]. As shown in Table 4, the transform in [3] outperforms all other transforms as it has the largest number of arithmetic operations. Compared to our transform, we notice a slight difference that varies from 0.2 dB (for Barbara) to 0.6 dB (for Airplane and Lena). Note that the cost savings in computation associated with our transform is 4 arithmetic operations without bit shifts.

We now compare the performance of the proposed transform with other transforms considered in this study. As the Table 4 illustrates, the results show a significant improvement over the transform PADCT [15] up to 2 dB. Moreover, even if our transform has the smallest number of operations, it outperforms the parametric transformation of [4].

Finally, a subjective evaluation is shown in Fig. 5 for the Lena test image. The image has been subjected to the JPEG-like compression experiment. Only 5 of the 64-transformed coefficients in each block were used reconstruct the image, all other coefficients were set to zero. This value is already considered in a previous work [10]. As may be noted, the reconstructed image using the proposed algorithm produces less blocking artifacts than the methods in [11, 15]. The quality of image reconstructed using transform in [11] diminished even though the number of operations has been reduced by two compared to the proposed one. As a result, the proposed transform offers a good compromise between performance and complexity.

Table 3 Running time comparison of the proposed transform against its original transform

Lena image size	256 × 256	512 × 512	1024 × 1024
Transform			
Proposed transform	0.0111	0.0446	0.317
Bas-2008 [3]	0.0120	0.0462	0.331
PADCT [15]	0.0116	0.0448	0.320
Scaled DCT [27]	0.0908	0.433	1.869

Table 4 Performance assessment of different transforms in terms of PSNR and compression ratio (CR)

Methods	Number of arithmetic operations		Test images					
			Lena	Baboon	Barbara	Boat	Airplane	Sailboat
Proposed transform	16 additions	CR	12.06	7.45	9.75	10.58	12.22	9.46
		PSNR	30.96	25.00	25.19	29.90	31.35	29.66
Bas-2008 [3]	18 additions and 2-bit shifts	CR	12.00	7.37	9.70	10.62	12.04	9.47
		PSNR	31.62	25.34	25.41	30.56	32.01	30.14
PADCT [15]	17 additions	CR	11.88	7.48	9.71	10.52	12.22	9.51
		PSNR	29.10	23.18	23.68	26.51	27.40	26.21
Bas-2011 [4] (a = 2)	18 additions and 2-bit shifts	CR	11.71	7.32	9.75	10.44	11.78	9.16
		PSNR	30.06	24.48	24.98	29.94	28.56	29.26
Bas-2011 [4] (a = 1)	18 additions	CR	12.06	7.35	9.67	10.48	12.00	9.40
		PSNR	30.92	25.06	25.31	29.94	30.60	29.24
Bas-2011 [4] (a = 0)	16 additions	CR	12.06	7.42	9.69	10.54	12.06	9.42
		PSNR	30.96	25.03	25.23	29.28	31.14	29.58
CB-2012 [11]	14 additions	CR	12.56	7.41	9.55	10.62	12.11	9.44
		PSNR	27.35	24.22	24.17	27.38	28.77	27.09
SDCT [14]	24 additions	CR	11.96	7.49	9.44	10.68	12.14	9.58
		PSNR	28.76	23.70	23.67	27.76	28.94	27.38
Scaled DCT [27]	29 additions and 5 multiplications	CR	11.96	7.54	9.50	10.54	12.15	9.44
		PSNR	34.98	26.18	26.37	33.26	35.32	32.56

**Fig. 5** The original (a) and reconstructed Lena image using, proposed transform (PSNR = 27.05 dB) (b), CB-2012 transform (PSNR = 24.77 dB) (c), PADCT (PSNR = 26.69 dB) (d), Bas-2011 {a = 2} (PSNR = 26.69 dB) (e) and SDCT (PSNR = 25.71 dB) (f)

7 Performance comparison with the Hadamard transform

In this section, we compare the compression performance of the proposed transform to the 8-points Hadamard transform [17]. For this purpose, we perform a similar experiment to that described in section 5, using the Hadamard transform [17], our transform and the conventional DCT transform. The performance comparison is provided in terms of PSNR, PEEN and SSIM as shown in Fig. 6. In comparison with the Hadamard transform requiring 24 additions, the results of our study clearly show a significant improvement in performance of the proposed transform with only 16 additions performed.

8 Conclusion

A novel orthogonal approximate DCT transform which appropriately approximates the conventional DCT of length 8×8 was proposed in this paper. The proposed transform was found by replacing shifts elements in the matrix transform reported in [3] with null elements. The resulting transform, requiring only 16 additions, outperforms existing approximate transforms having the same number of arithmetic operations and other transforms with a larger number of operations. Furthermore, an efficient algorithm has also been successfully developed at a lowest computation.

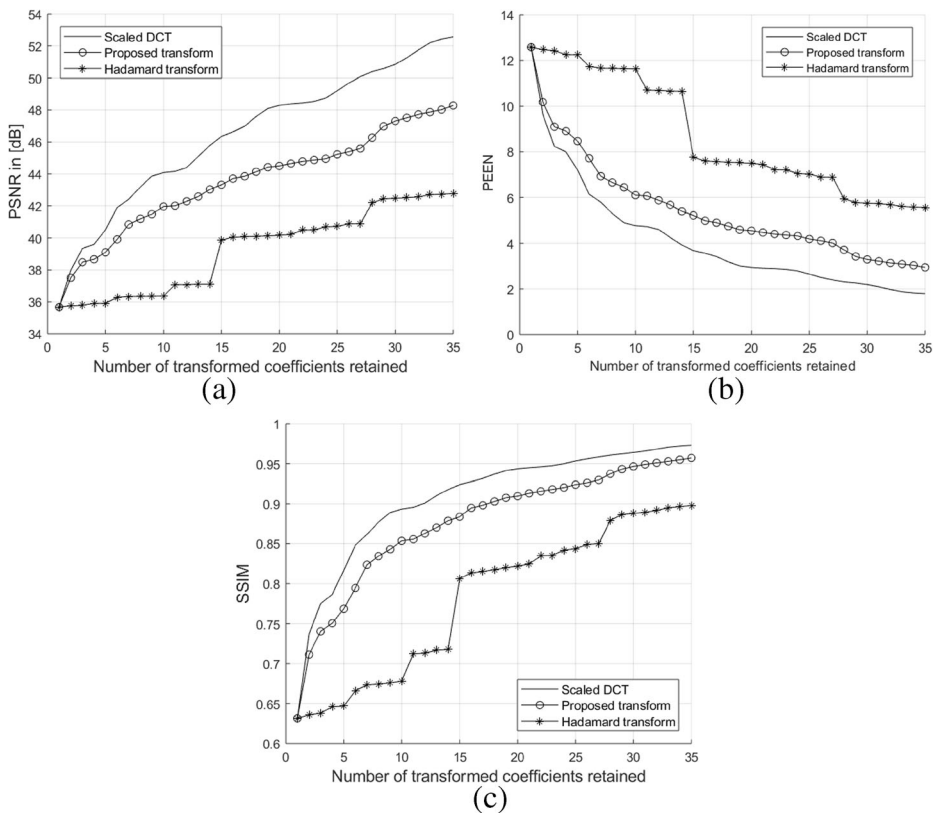


Fig. 6 Compression performance comparison between the proposed transform and the Hadamard transform in terms of (a) PSNR. (b) PEEN. (c) SSIM

References

1. Ahmed N, Natarajan T, Rao KR (1974) Discrete cosine transform [J]. *IEEE Trans Comput* 23(1):90–93
2. Almurib AFH, Kumar TN, Lombardi F (2018) Approximate DCT image compression using inexact computing [J]. *IEEE Trans Comput* 67(2):149–159
3. Bouguezel S, Ahmed MO, Swamy MNS (2008) Low complexity 8×8 transform for image compression [J]. *Electron Lett* 44(21):1249–1250
4. Bouguezel S, Ahmed MO, Swamy MNS (2011) A low-complexity parametric transform for image compression. [C]// proceeding of IEEE international symposium of circuits and systems (ISCAS): p 2145–2148
5. Bouguezel S, Ahmed MO, Swamy MNS (2013) Binary discrete cosine and Hartley transforms [J]. *IEEE Trans Circ and Syst I* 60(4):989–1000
6. Brahimi N, Bouguezel S (2011) An efficient fast integer DCT transform for images compression with 16 additions only [C]//proceeding of 7th international workshop on systems signal processing and their applications (WOSSPA), p 71–74.
7. Brahimi T, Melit A, Khelifi F (2009) An improved SPIHT algorithm for lossless image coding [J]. *Digital Signal Processing* 19(2):220–228
8. Brahimi T, Boubchir L, Fournier R, Nait-Ali A (2017) An improved multimodal signal image compression scheme with application to natural images and biomedical data [J]. *Multimed Tools Appl* 76(15):16783–16805
9. Brahimi T, Laouir F, Boubchir L, Ali-Chérif A (2017) An improved wavelet-based image coder for embedded greyscale and colour image compression [J]. *AEU Int J Electron Commun* 73:183–192
10. Cintra RJ, Bayer FM (2011) A DCT approximation for image compression [J]. *IEEE Signal Proc Lett* 18(10):579–582
11. Cintra RJ, Bayer FM (2012) DCT-like transform for image compression requires 14 additions only [J]. *Electron Lett* 48(15):919–921
12. Coelho DFG, Cintra RJ, Kulasekera S, Madanayake A, Dimitrov VS (2016) Error-free computation of 8-point discrete cosine transform based on the Loeffler factorization and algebraic integers [J]. *IET Signal Proc* 10(6):633–640
13. Ezhilarasi R, Venkatalakshmi K, Khanth BP (2018) Enhanced approximate discrete cosine transforms for image compression and multimedia applications [J]. *Multimed Tools Appl*:1–14
14. Haweel TI (2001) A new square wave transform based on the DCT [J]. *Signal Process* 81(11):2309–2319
15. Haweel RH, El-Kilani WS, Ramadan HH (2016) Fast approximate DCT with GPU implementation for image compression [J]. *J Vis Commun Image Represent* 40:357–365
16. Jridi M, Alfalou A, Meher PK (2015) A generalized algorithm and reconfigurable architecture for efficient and scalable orthogonal approximation of DCT [J]. *IEEE Trans Circ and Syst I* 62(2):449–457
17. Lee MH, Kaveh M (1986) Fast Hadamard transform based on a simple matrix factorization [J]. *IEEE Trans Acoust Speech Signal Process* 34(6):1666–1667
18. Liu S, Yu R, Huang H, Yang H (2014) Unified algorithms for computation of different points integer 1-D DCT/IDCT for the HEVC standard [C]// International Conference on Software Intelligence Technologies and Applications & International Conference on Frontiers of Internet of Things p. 207–211
19. Miano J (1999) Compressed image file formats JPEG, PNG, GIF, XBM, BMP. Addison Wesley Longman
20. Oliveira RS, Cintra RJ et al (2019) Low-complexity 8-point DCT approximation based on angle similarity for image and video coding [J]. *Multimed Syst Sign Process* 30(3):1363–1394
21. Potlur U, Madanayake A, Cintra R, Bayer F, Kulasekera S, Edirisuriya A (2014) Improved 8-point approximate DCT for image and video compression requiring only 14 additions [J]. *IEEE Trans Circ and Syst* 61(6):1727–1740
22. Rao KR, Kim DN, Hwang JJ (2014) Video coding standards. The Netherlands Springer:51–97
23. Rao K, Ramamohan K, Yip P (2014) Discrete cosine transform: algorithms, advantages, applications. Academic Press
24. Sayood K (2017) Introduction to data compression. Morgan Kaufmann, Elsevier
25. Senapati RK, Pati UC, Mahapatra KK (2010) A low complexity orthogonal 8×8 transform matrix for fast image compression [C]// proceeding of annual IEEE India conference (INCON): 1–4
26. Tamboli P, Shinde A (2015) A low complexity 8×8 DCT transform for image compression [J]. *IJAREEIE* 4(7):6185–6190
27. Wahid KA, Dimitrov VS, Jullien GA (2007) On the error-free realization of a scaled DCT algorithm and its VLSI implementation [J]. *IEEE Trans Circ and Syst II. Express Briefs* 54(8):700–704
28. Wallace GK (1992) The JPEG still picture compression standard [J]. *IEEE Trans Consum Electron* 38(1): xviii–xxxiv

29. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity [J]. *IEEE Trans Image Process* 13(4):600–612

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Nabila Brahimi¹ • Toufik Bouden² • Tahar Brahimi³ • Larbi Boubchir⁴

¹ NDT Laboratory, Electronic Department, University of Mohammed Seddik Benyahia, BP 98, 18000 Jijel, Algeria

² NDT Laboratory, Automatic Department, University of Mohammed Seddik Benyahia, BP 98, 18000 Jijel, Algeria

³ L2EI Laboratory, Electronic Department, University of Mohammed Seddik Benyahia, BP 98, 18000 Jijel, Algeria

⁴ LIASD Research Lab, Department of Computer Science, University of Paris 8, Paris, France